HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P. O. Box 272400
Fort Collins, Colorado 80527-2400

OIPE
JUN 0 5 2006
PATENT & TRADEMARK OFFICE

PATENT APPLICATION AF

ATTORNEY DOCKET NO. 10011537-1

IN THE
UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor(s): Robert D. Gardner

Confirmation No.:

Application No.: 10/033,102

Examiner: Syed J. Ali

Filing Date: 10/25/2001

Group Art Unit: 2195

Title: PROTECTION OF USER PROCESS DATA IN A SECURE PLATFORM ARCHITECTURE

Mail Stop Appeal Brief-Patents
Commissioner For Patents
PO Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL OF APPEAL BRIEF

Sir:

Transmitted herewith is the Appeal Brief in this application with respect to the Notice of Appeal filed on 03/29/2006.

The fee for filing this Appeal Brief is (37 CFR 1.17(c)) $500.00.

(complete (a) or (b) as applicable)

The proceedings herein are for a patent application and the provisions of 37 CFR 1.136(a) apply.

( ) (a) Applicant petitions for an extension of time under 37 CFR 1.136 (fees: 37 CFR 1.17(a)-(d) for the total number of months checked below:

( )	one month	$120.00
( )	two months	$450.00
( )	three months	$1020.00
( )	four months	$1590.00

( ) The extension fee has already been filled in this application.

(X) (b) Applicant believes that no extension of time is required. However, this conditional petition is being made to provide for the possibility that applicant has inadvertently overlooked the need for a petition and fee for extension of time.

Please charge to Deposit Account **08-2025** the sum of **$500.00**. At any time during the pendency of this application, please charge any fees required or credit any over payment to Deposit Account 08-2025 pursuant to 37 CFR 1.25. Additionally please charge any fees to Deposit Account 08-2025 under 37 CFR 1.16 through 1.21 inclusive, and any other sections in Title 37 of the Code of Federal Regulations that may regulate fees. A duplicate copy of this sheet is enclosed.

( ) I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, Alexandria, VA 22313-1450. Date of Deposit: May 31, 2006

OR

( ) I hereby certify that this paper is being transmitted to the Patent and Trademark Office facsimile number________ on ________

Number of pages:

Typed Name: **Joanne Bourguignon**

Signature: Joanne Bourguignon

Respectfully submitted,

Robert D. Gardner

By Robert W. Bergstrom

Robert W. Bergstrom

Attorney/Agent for Applicant(s)
Reg. No. **39,906**

Date: **May 31, 2006**

Telephone No.: **206.621.1933**

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re patent application of:

Applicant:	Robert W. Gardner
Application No.:	10/033,102
Filed:	October 25, 2001
Title:	Protection of User Process Data in a Secure Platform Architecture

Examiner:	Syed J. Ali
Art Unit:	2195
Docket No.:	10011537-1
Date:	May 29, 2006

APPEAL BRIEF

Mail Stop: Appeal Briefs – Patents
Commissioner of Patents and Trademarks
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This appeal is from the decision of the Examiner, in an Office Action mailed December 29, 2005, finally rejecting claims 1-26.

REAL PARTY IN INTEREST

The real party in interest is Hewlett-Packard Development Company, LP, a limited partnership established under the laws of the State of Texas and having a principal place of business at 20555 S.H. 249 Houston, TX 77070, U.S.A. (hereinafter "HPDC"). HPDC is a Texas limited partnership and is a wholly-owned affiliate of Hewlett-Packard Company, a Delaware Corporation, headquartered in Palo Alto, CA. The general or managing partner of HPDC is HPQ Holdings, LLC.

RELATED APPEALS AND INTERFERENCES

Applicant's representative has not identified, and does not know of, any other appeals or interferences which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

STATUS OF CLAIMS

Claims 1-26 are pending in the application. Claims 1-26 were finally rejected in the Office Action dated December 29, 2006. Applicant appeals the final rejection of claims 1-26, which are copied in the attached CLAIMS APPENDIX.

STATUS OF AMENDMENTS

No Amendment After Final is enclosed with this brief. The last Response was filed October 3, 2005.

SUMMARY OF CLAIMED SUBJECT MATTER

Overview

The current application is directed to a new approach to computer architecture that employs architectural features included in modern processor implementations, including the Intel IA-64®, or Itanium®, processors, to provide a secure platform kernel ("SPK") and secure platform global services ("SPGS") that run, at the most privileged privilege levels, above processor hardware and that, in turn, provide a virtual machine interface to one or more operating systems that run above the SPK and SPGS. The phrase "running above" the SPK and SPGS means that the operating system executes instructions at a privilege level less privileged than the privilege levels at which the SPK and SPGS may run. In one embodiment of the present invention, the SPK and SPGS may run at privilege levels 0 and 1, while the operating system runs at privilege level 2 or 3, with privilege level 0 representing the most privileged privilege level, and privilege level 3 representing the least privileged privilege level at which application programs execute. In other words, the current application is directed to a new computer architecture in which operating systems run above virtual machine interfaces that include privileged SPK and SPGS layers, rather than directly above the computer hardware, as currently available operating systems are designed to execute. In

this new architecture that represents an embodiment of the present invention, unlike in current computer architectures, operating systems cannot directly execute privileged instructions, and hardware resources protected at privilege levels more privileged than the privilege level at which an operating system executes cannot be accessed by the operating system that run above the virtual-machine interface unless explicitly allowed to do so by the SPK layer. This new architecture is discussed, in overview, in the current application beginning on line 11 of page 4.

Independent Claim 1

. Claim 1 is directed to a computer system (20 in Figures 1-3) comprising at least one processor (32 in Figure 1), a memory, a secure platform (36 in Figures 1-3) stored in the memory for controlling the processor and the memory, an operating system image (42 in Figures 1 and 3) stored in the memory for controlling the processor and the memory and operating on top of the secure platform, and an end-user application (44 in Figures 1 and 3) stored in the memory for controlling the processor and the memory and operating on top of the operating system image. In the computer system to which claim 1 is directed, the secure platform is configured to provide a secure partition within the memory for storing secret data associated with and accessible by the end-user application, the secure partition being inaccessible to the operating system and other tasks operating on top of the secure platform (current application, page 3, lines 12-15; page 32, lines 14-22). *In other words, claim 1 is directed to a computer system in which a secure platform, which operates at the highest available privilege levels, is interposed between computer-system hardware and the operating system, which executes at a privilege level less privileged than that at which the secure platform may execute, so that a portion of the computer system's memory can be associated with, and accessed by, an user application, but the portion of the computer system is not accessible to the operating system.*

Dependent Claims 2 – 11

Claim 2 further specifies that at least one processor has at least three execution privilege levels including a first privilege level, a second privilege level that is less privileged than the first privilege level, and a third privilege level that is less privileged than the second privilege level. Claim 3 further specifies that the end user application is configured to operate at the third privilege level as an unprivileged task, the operating system image is

configured to operate at the second privilege level as an unprivileged task, and at least a first portion of the secure platform is configured to operate at the first privilege level as a privileged task. Claim 4 further specifies that the first portion of the secure platform is a secure platform kernel (SPK). Claim 5 further specifies that the SPK performs security critical services including memory services. Claim 6 further specifies that the security critical services performed by the SPK further include process services, cryptographic services, and exception handling. Claim 7 further specifies that at least one processor includes protection key registers configured to hold protection keys, which the secure platform employs to control access to security critical structures. Claim 8 further specifies that the security critical structures include the secure partition. Claim 9 further specifies that the secure partition includes at least one memory page. Claim 10 further specifies that the security critical structures include the end-user application. Claim 11 further specifies that the end-user application includes a secure process indicator for indicating that the end user application is to be treated as a secure process.

Independent Claim 12

Claim 12 is directed to a method of controlling a computer system (20 in Figures 1-3) that includes a memory and at least one processor having at least three execution privilege levels, the execution privilege levels including a first privilege level, a second privilege level that is less privileged than the first privilege level, and a third privilege level that is less privileged than the second privilege level, that at least one processor also having protection key registers is configured to hold protection keys that are employed to control access to security critical structures. The method to which claim 12 is directed includes steps of: operating a secure platform kernel (SPK) (36 in Figures 1-3) at the first privilege level as a privileged task; operating an operating system (42 in Figures 1 and 3) at the second privilege level as an unprivileged task; operating an end-user application (44 in Figures 1 and 3) at the third privilege level as an unprivileged task; allocating a portion of the memory for use by the end-user application; associating a first protection key value with the allocated memory portion; and inserting the first protection key value in one of the protection key registers only when instructions of the end-user application are being executed, thereby allowing the end-user application to access the allocated memory portion and preventing other tasks operating at the second and the third privilege levels from accessing the allocated memory portion (current application, page 3, lines 12-15; page 32, lines 14-22). *In other*

words, claim 12, like claim 1, is directed to a computer system in which a secure platform, which operates at the highest available privilege levels, is interposed between computer-system hardware and the operating system, which executes at a privilege level less privileged than that at which the secure platform may execute, so that a portion of the computer system's memory can be associated with, and accessed by, an user application, but the portion of the computer system is not accessible to the operating system.

<u>Dependent Claims 13 – 17</u>

Claim 13 includes the additional steps of: monitoring execution of instructions of the end-user application; and flushing the first protection key value from the protection key registers when execution of the end user application instructions stops. Claim 14 includes the additional step of reinserting the first protection key value in one of the protection key registers when execution of the end-user application instructions resumes. Claim 15 further specifies that allocating a portion of the memory is performed by the SPK. Claim 16 further specifies that the first protection key value is inserted in one of the protection key registers by the SPK. Claim 17 includes the additional step of associating a second protection key with the end-user application to prevent unauthorized modification.

<u>Independent Claim 18</u>

Claim 18 is directed to a computer system (20 in Figures 1-3) that includes at least one processor (32 in Figure 1) having at least three execution privilege levels, the execution privilege levels including a first privilege level, a second privilege level that is less privileged than the first privilege level, and a third privilege level that is less privileged than the second privilege level, a memory, an end-user application (44 in Figures 1 and 3) stored in the memory for controlling the processor and the memory, the end-user application configured to operate at the third privilege level as an unprivileged task, an operating system (42 in Figures 1 and 3) stored in the memory for controlling the processor and the memory, the operating system configured to operate at the second privilege level as an unprivileged task, and a secure platform (36 in Figures 1-3) stored in the memory for controlling the processor and the memory, at least a first portion of the secure platform configured to operate at the first privilege level as a privileged task, the secure platform configured to provide a secure storage area in the memory for data associated with the end-user application, the secure storage area accessible to a properly authenticated user of the end-user application, the

secure storage area inaccessible to other tasks operating at the second and third privilege levels, including the operating system (current application, page 3, lines 12-15; page 32, lines 14-22). *In other words, claim 18, like claims 1 and 12, is directed to a computer system in which a secure platform, which operates at the highest available privilege levels, is interposed between computer-system hardware and the operating system, which executes at a privilege level less privileged than that at which the secure platform may execute, so that a portion of the computer system's memory can be associated with, and accessed by, an user application, but the portion of the computer system is not accessible to the operating system.*

Dependent Claims 19-25

Claim 19 further specifies that the first portion of the secure platform is a secure platform kernel (SPK). Claim 20 further specifies that the SPK performs security critical services including memory services. Claim 21 further specifies that the security critical services performed by the SPK further include process services, cryptographic services, and exception handling. Claim 22 further specifies that at least one processor includes protection key registers configured to hold protection keys, which the secure platform employs to control access to security critical structures. Claim 23 further specifies that the security critical structures include the secure storage area. Claim 24 further specifies that the secure storage area includes at least one memory page. Claim 25 further specifies that the security critical structures include the end-user application.

Independent Claim 26

Claim 26 is directed to a computer readable medium containing a secure platform (36 in Figures 1-3), an operating system image (42 in Figures 1 and 3), and an end-user application (44 in Figures 1 and 3) configured for controlling a computer system to perform a method, the computer system having a memory and at least one processor.

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1. The rejection of claims 1-2, 11, and 26 as being anticipated under 35 U.S.C. § 102(e) by McNabb et al., U.S. Patent No. 6,289,462 ("McNabb").

2. The rejection of claims 3-6 and 18-21 under 35 U.S.C. § 103(a) as being unpatentable

over McNabb.

3. The rejection of claims 7-10, 12-17, and 22-25 under 35 U.S.C. § 103(a) as being unpatentable over McNabb in view of Quach et al., U.S. Patent No. 6, 654, 909 ("Quach").

ARGUMENT

Claims 1 -26 are pending in the current application. In an Office Action dated December 29, 2005 ("Office Action"), the Examiner finally rejected claims 1-2, 11, and 26 under 35 U.S.C. § 102(e) as being anticipated by McNabb et al., U.S. Patent No. 6,289,462 ("McNabb"), finally rejected claims 3-6 and 18-21 under 35 U.S.C. § 103(a) as being unpatentable over McNabb, and finally rejected claims 7-10, 12-17, and 22-25 under 35 U.S.C. § 103(a) as being unpatentable over McNabb in view of Quach et al., U.S. Patent No. 6,654, 909 ("Quach"). Applicant's representative respectfully traverses these 35 U.S.C. § 102 and 35 U.S.C. § 103 rejections.

ISSUE 1

1. Whether claims 1-2, 11, and 26 are anticipated under 35 U.S.C. § 102(e) by McNabb et al., U.S. Patent No. 6,289,462 ("McNabb").

As discussed above, embodiments of the current invention are directed to computer systems that feature a new architecture in which a secure platform kernel ("SPK") and secure platform global services ("SPGS") are interposed between the hardware components of the computer system and the operating system that runs on the computer system above the SPK and SPGS. This new architecture allows a portion of memory within the computer system to be allocated by the SPK to an end-user application running above the operating system in a way that the end-user application may access the allocated memory, but the operating system, running below the user application, cannot access the allocated memory. This architecture radically departs from current computer-system architectures, in which the operating system can access each and every hardware, firmware, software, and data resource of the computer system, since in current computer-system architectures, the operating systems are designed run directly above the hardware and to run at the highest available privilege level. This new computer-system architecture is made possible by the

Intel Itanium® processor architecture, which provides hardware mechanisms that allow certain, privileged instructions to be executed only by processes running at a highest privilege level, privilege level 0, while providing sufficient privilege levels to accommodate entities more privileged than the operating system, the operating system, and least-privileged end-user applications. In addition, the Itanium architecture allows memory to be protected, based on privilege levels and protection keys, at page granularity. When the operating system of a computer system runs at privilege level 2, while an SPK that represents one embodiment of the present invention runs at the more privileged privilege levels 0 and 1, the SPK can allocate memory and manage that cannot be accessed by the operating system. Moreover, since the SPK, and only the SPK, can access interrupt vectors and other privileged registers and instructions, the SPK can arrange to be invoked when any process attempts to access protected memory. The SPK can then allow access to an end-user application, on an access-by-access basis, to SPK-protected memory, while denying access to the operating system. The Itanium architecture provides a variety of complex privilege-level-controlled and protection-key-controlled instructions and resources that allow the SPK to be interposed between the hardware and the operating system, and was specifically designed to do so, although, in general, the Itanium provides a strict privilege hierarchy, in which a more privileged process can access all equally and less privileged instructions and resources. The current invention provides a way for a resource - in the described system, a portion of physical memory - to be accessible by a least-privileged end-user application without being accessible to a more-privileged operating system.

The ability to provide protected memory to end-user applications that cannot be accessed by the underlying operating system is an important feature of the new computer-system architecture to which claim 1 is directed. Claim 1 specifically states that "the secure platform is configured to provide a secure partition within the memory for storing secret data associated with and accessible by the end-user application, the secure partition being inaccessible to the operating system and other tasks operating on top of the secure platform." In rejecting claim 1, the Examiner states that this feature of the new computer-system architecture to which claim 1 is directed is taught in McNabb on lines 20-24 of column 4 and lines 7-17 and 52-61 of column 17. McNabb teaches nothing of the sort.

First, as discussed in the first paragraph of the detailed description of the preferred embodiment of McNabb, on lines 11-20 of column 7, McNabb details exactly those components which McNabb combines to create a trusted server:

> The present invention is a fully integrated software foundation for secure business processes. It incorporates a variety of security technologies into a seamless, secure system. Its components and modifications comprise: operating system security enhancements, network packet management, modifications, upgrade/downgrade enforcer (UDE), security gate, trusted administration utilities, enhanced secure shell (enhanced SSH), authentication module, secure CGI module, network layer encryption (VPN), and usage of Secure Socket Layer Encryption (SSL).

Those familiar with computer-system architecture, operating systems, and computer security will immediately recognize that the lowest-level component mentioned in this list of components and enhancements is the operating system. Network package management, upgrade/downgrade enforcers, the security gate, trusted administration utilities, secure shell, the authentication module, the secure CGI module, the network layer encryption, and secure socket layer encryption are all either operating-system modules and utilities or higher-level programs that run above the operating system. There is no teaching, or even suggestion, in McNabb that McNabb's trusted server incorporates any component or component enhancement at a level below the operating system.

In the paragraph beginning on line 54 of column 8, McNabb discusses modifications of the operating system used to create McNabb's trusted server:

> The key components discussed below for the trusted operating system are as follows: 1) processes; 2) file system objects (including devices, directories, files, etc.); and 3) interprocess communication messages (including packets, shared memory, etc.). *On a standard system, each of these has various security attributes, which are created, managed, and used by the OS itself. When a process attempts to access a file system object, the OS compares various attributes of the process with attributes of the object, and allows or denies access. When a process sends or receives communication messages, the OS verifies that the process is allowed to send and/or receive the message. When objects are created, such as when a file is created or when a message is generated, the OS is responsible for ensuring that the proper attributes are attached to the new object.*
>
> *The trusted server system has extended this standard mechanism in two ways: by attaching additional security attributes to each of the OS components, and by extending the security checks to use the new attributes. A security is the "sensitivity label" 202 or SL.* Fig. 2 shows the extended attributes applied to files while Fig. 3 shows the attributes

> that are applied to the processes or packets that are processed on the system. Fig. 4 shows the attribute types in a tabular form that are part of this structure. Unlike standard security mechanisms, the SL 202 is designed to be mandatory, i.e., not under control of the user. *This means that though a user may own a file, he would be unable to make it, or its contents or even a copy, accessible to a user that was not previously authorized to have such information.* SLs can be related in several ways: 1) they can be equal, 2) one can be "greater" (dominates) than another, and 3) they can be "disjoint" (meaning neither is greater than the other). (emphasis added)

It is clear from the above-quoted portion of McNabb that McNabb's trusted server relies on operating-system enhancements that include adding a new type of "sensitivity label" ("SL") security attribute to the attributes commonly associated by an operating system with objects, such as files, and by extending operating-system security checks to check the new "sensitivity label" attribute. As explicitly stated in the above-quoted portion of McNabb, these new attributes prevent users from making user-owned objects accessible to other users who lack appropriate authorization. Thus, the "sensitivity label" attributes are supplied and controlled by the operating system in order to control access to objects controlled by the operating system on behalf of end users. McNabb does not describe a mechanism for protecting a computer-system resource from the enhanced operating system.

Beginning on line 40 of column 9, McNabb details this process, explicitly stating that it is the operating system that provides, controls, and checks the security attributes:

> When a process makes this request, *the OS compares the attributes of the process to the attributes of the file where the program is stored to see if the process will be allowed to run the program.* *<u>The additional security attributes of the trusted server system are used by the OS to allow greater control over which programs are available.</u>* In addition to the standard user and group identifiers, the trusted server system has added an "authorization database" in the OS that is used to see if the user running the process can access or execute the requested program. (emphasis added)

In similar vein, McNabb details operating-system control and checking of security attributes beginning on line 26 of column 10:

> When a process sends information via a communications channel or network, the trusted server has modified the OS so that the process's

> security attributes are attached to the packet. If the communication is with another process on the same machine, the security attributes can be checked before the destination process is given access to the packets. *Thus the OS can enforce security on processes based on various security attributes, such as SLs.* (emphasis added)

Returning to the Examiner's anticipation rejection, the Examiner cites the following portion of column 4 as teaching "the secure platform is configured to provide a secure partition within the memory for storing secret data associated with and accessible by the end-user application:"

> What is desired therefore is a system where these components are fully integrated to provide a secure platform for network services, where users can install the system and immediately begin taking advantage of its security features, installing applications and servers into protected partitions.

This quoted portion of McNabb simply states that users can install McNabb's trusted server system and take advantage of its security features, by installing applications and servers into protected partitions. McNabb obviously uses the term "secure platform" to mean the enhanced operating system and additional utilities and programs that provide a platform for a network services application. The term "secure platform kernel" in the current application is defined, by contrast, as an entity interposed between the computer system hardware and operating system that executes at higher privilege than the operating system. This quoted portion of McNabb mentions nothing about secret data associated with and accessible by end-user applications. Instead, this quoted section refers to installing the applications themselves into protected partitions. The protections provided by the protected partitions are not specified, but in Unix system, protected partitions are an operating system created and controlled construct. Even if this above-quoted portion of McNabb could be misinterpreted to be construed as teaching of storing secret data in a secure partition that can be accessed by, and associated with, an end-user applications, *which it does not*, the next cited portions of McNabb can in no way be construed to teach that the secure partition accessible to, and associated with, an end-user application is not accessible to the operating system. These cited portions are provided below:

> Fig14 shows a web server used to display static web pages and to provide connectivity to back-end applications. *The information is*

> *passed through a security gate at step 160. The data in a back-end application resides in its own compartment (has a unique sensitivity label), ensuring that the back-end application will be accessible only through the security gate 8.* With this approach, any attempts to transmit information directly from the Internet interface to the internal interface, or vice versa, will be defeated.
>
> It is not always possible for a server to predict the IP address that will be used by a specific user. (emphasis added)

> The ability to partition a network server is a key component of the trusted operating system of the present invention in providing the level of assurance needed to support critical network and transaction servers. *Referring to Fig. 12, processes, files, and other resources that have the same sensitivity label are said to be in the same compartment or partition 11. Programs, data, and network interfaces can be split into separate, isolated partitions with restricted access between them. For example back end applications related to specific functions may be stored in separate application compartments 12,13 that are isolated from each other and from the externally accessible compartment 11 that is accessible by the general public.* (emphasis added)

The first above-cited portion of McNabb discusses a web server that provides connectivity to back-end applications. Web servers are application programs that run above an operating system. This portion of McNabb discusses information being passed through a security gate (160 in Figure 14). The data in the back-end application resides in its own compartment, which means, as explicitly stated by McNabb, that the data has a unique sensitivity label, ensuring that the back-end application will be accessible only through the security gate. Beginning on line 39 of column 11, McNabb describes a security gate:

> *A Security Gate 8 (see Fig. 1) program is a special program that allows limited communication between two processes, or between a process and a network interface.* Its configuration file specifies the source SL and port, along with the destination SL and port. In a typical configuration, the source and destination SLs are disjoint, meaning that the two ends cannot interact in any way. *The Security Gate program is given an SL that is greater than both. It is also given an SL range that includes both endpoints, along with the privilege to override the SL security checks, but limited to only the SLs within its range.* (emphasis added)

Note that the security gate employs the "sensitivity label" special attribute. As already discussed above, the "sensitivity label" ("SL") is an operating-system provided and operating-system controlled security attribute. Thus, the security gate is based on operating-system

enhancements that allow the operating system to decide whether or not to allow access to computer-system resources controlled by the SL. In other words, the operating system has full access to any computer-system resource controlled by "sensitivity label" attributes, since it is the operating system that decides to whom access is provided. Moreover, web servers, security gates, and all the other components mentioned in the first above-cited paragraph execute above the operating system, at privilege levels less privileged than those of the operating system.

The second of the two above-quoted, cited portions of McNabb, provided before the above-quoted description of security gates, discuss sensitivity-label-controlled access to partitions. This paragraph discusses storing different back-end application programs in separate application compartments that are isolated from one another and from the general public. Again, because this isolation and compartmentalization is achieved by using sensitivity labels, the operating system of McNabb's trusted server necessarily has access to these partitions and compartments. Again, as discussed above, it is McNabb's modified operating system that creates and controls sensitivity labels, and that decides which processes may access computer-system components protected by sensitivity levels. Nothing in these cited portions of McNabb refer to anything that is inaccessible to the operating system.

Thus, there is absolutely no teaching or suggestion in McNabb for memory, or any other computer-system resource, that can be accessed by an end-user application, but not by the operating system. The Examiner has not cited such a teaching or suggestion, McNabb contains no such teaching or suggestion, and, because McNabb discloses a standard computer architecture in which the operating system can access any and all computer-system resources, McNabb's trusted server cannot possibly provide memory accessible to an end-user application, but not to the operating system. McNabb clearly states that McNabb's trusted server is implemented by operating-system enhancements and introducing various system and application programs that operate above the operating system.

As stated in MPEP § 2131, and in many different Federal circuit opinions:

> "[a] claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." *Verdegaal Bros. v. Union Oil CO. of California*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). "The identical invention must be shown in as complete detail as is contained in ... claim." *Richardson v. Suzuki Motor Co.*, 868 F.2d 1226, 1236, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989).

Claim 1 cannot be possibly anticipated by McNabb, since McNabb fails to teach, disclose, mention, or even suggest a computer resource accessible by an end-user application but inaccessible to the operating system. Furthermore, McNabb does not once mention or suggest any type of component or executable process analogous to, or similar to, Applicant's clearly claimed secure kernel platform. Applicant's secure-kernel platform executes privileged instructions that cannot be executed by the operating system, and is interposed between the computer-system hardware and the operating system. There is nothing related to, or even remotely similar to, Applicant's secure-kernel platform disclosed or suggested in McNabb. For this reason, claim 26, which explicitly claims a secure platform, cannot possibly be anticipated by McNabb. Moreover, none of claims 2 and 11, which depend from claim 1, can be anticipated by McNabb, for the same reason.

The Examiner's rationale for rejection of claims 2 and 11 include further misrepresentations of McNabb. For example, the Examiner states that the claim language "wherein the at least one processor has at least three execution privilege levels, including a first privilege level, a second privilege level that is less privileged than the first privilege level, and a third privilege level that is less privileged than the second privilege level" on lines 50-65 of column 12. Those lines are provided below:

> Other restrictions implemented to secure the operating system of the present invention require the segmentation of the superuser (root) privilege, where the root account is not permitted to execute all of the processes associated with system administration. Instead this privilege is broken down into many smaller privileges. Thus the backup program may be able to read any file, but it cannot be exploited to shut down the system, modify files, or send random network packets. The use of many limited capabilities instead of a single all-powerful mechanism is called in the prior art, the least privilege principle.

Please note that the above-quoted portion of McNabb, cited by the Examiner as teaching execution privilege levels provided by a processor, does not once mention processors or processor architectures. Instead, this portion of McNabb discusses operating-system superuser (root) privileges, root accounts, and other operating-system-level features. Hardware processors do not implement superuser privilege, root accounts, and other such operating-system features. Processors do not implement files and network packets. All of these are operating-system constructs that are controlled by an operating-system executable program. By contrast, processors provide a relatively small set of low-level instructions and

registers that allow for the execution of software programs compiled into, or interpreted as, sequences of low-level instructions. Nothing in the cited portion of McNabb is at all related to processor execution privilege levels.

ISSUE 2

2. Whether claims 3-6 and 18-21 are unpatentable over McNabb under 35 U.S.C. § 103(a).

Claims 3-6 depend from claim 1. Claim 18 is an independent claim that, like claim 1, as discussed above, is directed to a computer system in which a portion of memory can be allocated by a secure kernel for association with, and access by, an end-user application without be accessible to the operating system. Claim 18 even more specifically claims that Applicant's secure platform executes at a more privileged privilege level than the operating system. Claims 19-21 depend from claim 18.

As discussed above with respect to the Examiner's 35 U.S.C. § 102(e) anticipation rejection of claim 1, McNabb does not teach, disclose, mention, or even suggest any type of executable component interposed between a computer system's hardware and the computer system's operating system, such as Applicant's clearly claimed secure kernel platform. Moreover, as discussed above with respect to the 35 U.S.C. § 102 rejections, McNabb discloses a trusted server implemented by enhancing an existing Unix operating system and by adding higher-level programs and utilities. A key enhancement in McNabb's system is an additional security attribute referred to by McNabb as the "sensitivity level" ("SL"). As discussed with respect to the 35 U.S.C. § 102 anticipation rejections, because McNabb's enhanced operating system creates, controls, and checks these security attributes, McNabb's operating system necessarily has access to all computer-system resources protected by these security attributes. It is McNabb's enhanced operating system that determines which processes may access any particular computer-system resource, including files and memory, within McNabb's trusted server Therefore, McNabb does not, and cannot possibly, teach or suggest a portion of memory that is accessible to, and associated with, an end-user application, but not the operating system. As discussed above, currently available operating systems, including the Unix operating system, execute at the highest privilege level directly above the hardware level, and control access to all computer-system resources.

According to MPEP § 2143:

To establish a *prima facie* case of obviousness, three basic criteria must be

> met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations.

Because McNabb does not teach, disclose, mention, or suggest any computer resource accessible to an end-user application, but not to the operating system, and because McNabb does not teach, disclose, mention, or suggest any entity more privileged than the operating system, such as Applicant's clearly and distinctly claimed SPK, McNabb cannot possibly make obvious the secure-kernel-platform-based computer-system architecture claimed in claim 18, or any claim depending from claim 18 or from claim 1.

In the 35 U.S.C. § 103 obviousness-type rejections, the Examiner appears to misconstrue Applicant's "secure platform kernel ("SPK"). For example, in the rejection of claim 4, the Examiner states that McNabb teaches the invention as claimed, including the computer system of claim 3, wherein the first portion of the secure platform is a secure platform kernel, citing lines 60-65 of column 10 and lines 3-9 of column 11. These cited portions of McNabb are provided below:

> In the preferred embodiment of the Unix implementation, the trusted server system comprises a computer executing the kernel process of the present invention where in addition to the default inode structure information, a link is included to retrieve previously stored attribute label information related to the file.
>
> The kernel is adapted to retrieve this storage. This label may enable the processor to retrieve the file stored at that reference location directly if the user's base authority and permissions permit the execution of the file and if the extended attributes in the system authorizes the user to access that level of information.

These portions of McNabb refer to the Unix operation-system kernel, well known to anyone familiar with operating systems. The Unix operating-system kernel is merely one portion of the Unix operating system. Inode structures are Unix operating system constructs, for example. Files are also operating-system constructs, and are created, deleted, accessed, and modified using operating-system-provided utilities and routines. The Unix kernel not related to Applicant's clearly disclosed and claimed "secure platform kernel," which is not part of an operating system and, more importantly, which is more privileged than the operating systems

that run above it. Only an entity more privileged than an operating system can allocate computer-system resources, including memory, inaccessible to the operating system. Note that the second of the above-quoted portions of McNabb cited by the Examiner teach that the kernel is adapted to retrieve label information. This is also a direct reference to the Unix operating system kernel, explicitly stated by McNabb to be modified in order to implement his trusted server.

ISSUE 3

3. Whether claims 7-10, 12-17, and 22-25 are unpatentable over McNabb in view of Quach et al., U.S. Patent No. 6, 654, 909 ("Quach") under 35 U.S.C. § 103(a).

Quach explicitly states that Quach's patent is related "to the design of highly reliable microprocessors and more specifically to the use of a dedicated state machine that periodically checks the validity of critical processor resources." In other words, Quach discloses a feature of a processor architecture, and the error-detection state machine disclosed by Quach is shown, in Figure 1, as a component of a processor. The current application is not directed to processor architectures. Quach is unrelated to Applicant's claimed invention. Claims 7-10 depend from claim 1. Claim 12 is independent, and claims 13-17 depend from claim 12. Like claim 18, claim 12 explicitly states that Applicant's secure platform executes at a privilege level more privileged than that at which the operating system executes. Claims 22-25 depend from claim 18. As discussed above, all of the independent claims of the current application are directed to a computer system embodying a new computer-system architecture in which a portion of memory may be allocated for access by, and association with, an end-user application, but the portion of memory cannot be accessed by the operating system. As discussed above, McNabb neither teaches, discloses, mentions, or even suggests any kind of computer resource that can be accessed by an end-user application, but not by the operating system. McNabb discloses a trusted server implemented by modifying the currently available Unix operating system and adding additional operating-system utilities and other executable programs that execute above the operating system. Nowhere in McNabb does McNabb teach, disclose, mention, or suggest any kind of security attribute or access control that is not implemented within the operating system or operating-system utilities. McNabb does not once state or suggest that a computer resource within McNabb's trusted server cannot be accessed by McNabb's enhanced operating system. No combination

of McNabb and Quach can possibly make obvious any of the independent claims of the current application, or claims that depend from them.

CONCLUSION

In the Examiner's response to Applicant's arguments, the Examiner states that "Applicant asserts that McNabb is '*unrelated to the current application*.'" The Examiner further states that "Applicant argues that the claimed invention is a '*new computer architecture*' and that McNabb improves '*a currently available operating system*' and thus does not teach or suggest a '*new computer architecture*.'" That is not at all what Applicant asserts. Instead, Applicant asserts that the currently claimed invention, directed to a new computer architecture, provides for allocation of memory for association with, and access by, and end-user application that cannot be accessed by the operating system of the computer system. This is a novel type of memory allocation that is impossible on currently available operating systems designed to run directly above the computer hardware, to run at the highest privilege levels, and to therefore control and have access to all computer-system resources, including all of physical memory. McNabb's modified Unix operating system is a standard operating system designed to execute at the highest privilege levels and to therefore be able to access all computer-system resources. Had Applicant merely argued newness, as suggested by the Examiner, the Examiner's rejections might be well founded. But that is not what Applicant argued previously, and not what Applicant argues above. Applicant instead points to a clearly and distinctly claimed type of memory allocation made possible by Applicant's new computer architecture that is nowhere taught, disclosed, mentioned, or suggested in McNabb and that cannot be implemented using a traditional computer architecture in which the operating system can access any and all computer-system resources.

The Examiner continues to state, in the Examiner's response to Applicant's arguments, that "Applicant has mischaracterized McNabb and overlooked features relating to the 'secure platform.' The 'secure platform' is not the disclosed operating system enhancements, as Applicant suggests throughout the arguments, but the security features that are coupled to the operating system enhancements to make the entire system more secure." The Examiner has failed to understand either McNabb or the current application. No security features coupled to an operating system can provide for allocation of memory that can be accessed by an end-user application but not by the operating system. Such memory can only

be provided by an entity more privileged than an operating system, such as Applicant's clearly claimed SPK, and McNabb does not teach or suggest any such entity. Moreover, McNabb nowhere teaches, discloses, mentions, or suggests any type of computer system resource that cannot be accessed by the operating system. The portions of McNabb cited by the Examiner discuss protection of users from one another, and from the general public, but do not disclose, mention, or suggest a computer system resource that cannot be accessed by the operating system. Applicant's secure-kernel platform is explicitly claimed, in each independent claim, to be separate from, and to execute below or, in other words, at a higher privilege level, than the operating system. McNabb teaches nothing that executes at a greater privilege level than the enhanced operating system McNabb uses to implement McNabb's trusted server. Although McNabb may well provide well-regarded security features coupled to operating-system enhancements, those security features are entirely irrelevant to the currently claimed invention and do not provide memory accessible to an end-user application that is not also accessible to the operating system. Instead, McNabb's security features provide for more secure network-service applications.

Applicant respectfully submits that all statutory requirements are met and that the present application is allowable over all the references of record. Therefore, Applicant respectfully requests that the present application be passed to issue.

Respectfully submitted,
Robert W. Gardner
OLYMPIC PATENT WORKS PLLC

By Robert W. Bergstrom
Robert W. Bergstrom
Registration No. 39,906

Olympic Patent Works PLLC
P.O. Box 4277
Seattle, WA 98104
206.621.1933 telephone
206.621.5302 fax

CLAIMS APPENDIX

1. (original) A computer system comprising:

at least one processor;

a memory;

a secure platform stored in the memory for controlling the processor and the memory;

an operating system image stored in the memory for controlling the processor and the memory and operating on top of the secure platform;

an end user application stored in the memory for controlling the processor and the memory and operating on top of the operating system image; and

wherein the secure platform is configured to provide a secure partition within the memory for storing secret data associated with and accessible by the end user application, the secure partition being inaccessible to the operating system and other tasks operating on top of the secure platform.

2. (original) The computer system of claim 1, wherein the at least one processor has at least three execution privilege levels including a first privilege level, a second privilege level that is less privileged than the first privilege level, and a third privilege level that is less privileged than the second privilege level.

3. (original) The computer system of claim 2, wherein the end user application is configured to operate at the third privilege level as an unprivileged task, the operating system image is configured to operate at the second privilege level as an unprivileged task, and at least a first portion of the secure platform is configured to operate at the first privilege level as a privileged task.

4. (original) The computer system of claim 3, wherein the first portion of the secure platform is a secure platform kernel (SPK).

5. (original) The computer system of claim 4, wherein the SPK performs security critical services including memory services.

6. (original) The computer system of claim 5, wherein the security critical services performed

by the SPK further include process services, cryptographic services, and exception handling.

7. (original) The computer system of claim 1, wherein the at least one processor includes:

protection key registers configured to hold protection keys, which the secure platform employs to control access to security critical structures.

8. (original) The computer system of claim 7, wherein the security critical structures include the secure partition.

9. (original) The computer system of claim 8, wherein the secure partition includes at least one memory page.

10. (original) The computer system of claim 7, wherein the security critical structures include the end user application.

11. (original) The computer system of claim 1, wherein the end user application includes a secure process indicator for indicating that the end user application is to be treated as a secure process.

12. (original) A method of controlling a computer system, the computer system including a memory and at least one processor having at least three execution privilege levels, the execution privilege levels including a first privilege level, a second privilege level that is less privileged than the first privilege level, and a third privilege level that is less privileged than the second privilege level, the at least one processor also having protection key registers configured to hold protection keys that are employed to control access to security critical structures, the method comprising:

operating a secure platform kernel (SPK) at the first privilege level as a privileged task;

operating an operating system at the second privilege level as an unprivileged task;

operating an end user application at the third privilege level as an unprivileged task;

allocating a portion of the memory for use by the end user application;

associating a first protection key value with the allocated memory portion;

inserting the first protection key value in one of the protection key registers only when

instructions of the end user application are being executed, thereby allowing the end user application to access the allocated memory portion and preventing other tasks operating at the second and the third privilege levels from accessing the allocated memory portion.

13. (original) The method of claim 12, and further comprising:

monitoring execution of instructions of the end user application; and

flushing the first protection key value from the protection key registers when execution of the end user application instructions stops.

14. (original) The method of claim 13, and further comprising:

reinserting the first protection key value in one of the protection key registers when execution of the end user application instructions resumes.

15. (original) The method of claim 12, wherein the allocating a portion of the memory is performed by the SPK.

16. (original) The method of claim 12, wherein the first protection key value is inserted in one of the protection key registers by the SPK.

17. (original) The method of claim 12, and further comprising:

associating a second protection key with the end user application to prevent unauthorized modification.

18. (original) A computer system comprising:

at least one processor having at least three execution privilege levels, the execution privilege levels including a first privilege level, a second privilege level that is less privileged than the first privilege level, and a third privilege level that is less privileged than the second privilege level;

a memory;

an end user application stored in the memory for controlling the processor and the memory, the end user application configured to operate at the third privilege level as an unprivileged task;

an operating system stored in the memory for controlling the processor and the

memory, the operating system configured to operate at the second privilege level as an unprivileged task;

a secure platform stored in the memory for controlling the processor and the memory, at least a first portion of the secure platform configured to operate at the first privilege level as a privileged task, the secure platform configured to provide a secure storage area in the memory for data associated with the end user application, the secure storage area accessible to a properly authenticated user of the end user application, the secure storage area inaccessible to other tasks operating at the second and third privilege levels, including the operating system.

19. (original) The computer system of claim 18, wherein the first portion of the secure platform is a secure platform kernel (SPK).

20. (original) The computer system of claim 19, wherein the SPK performs security critical services including memory services.

21. (original) The computer system of claim 20, wherein the security critical services performed by the SPK further include process services, cryptographic services, and exception handling.

22. (original) The computer system of claim 18, wherein the at least one processor includes:

protection key registers configured to hold protection keys, which the secure platform employs to control access to security critical structures.

23. (original) The computer system of claim 22, wherein the security critical structures include the secure storage area.

24. (original) The computer system of claim 23, wherein the secure storage area includes at least one memory page.

25. (original) The computer system of claim 22, wherein the security critical structures include the end user application.

26. (original) A computer readable medium containing a secure platform, an operating system image, and an end user application configured for controlling a computer system to perform a method, the computer system having a memory and at least one processor, the method comprising:

controlling the processor and the memory with the secure platform;

controlling the processor and the memory with the operating system image which operates on top of the secure platform;

controlling the processor and the memory with the end user application which operates on top of the operating system image; and

creating a secure partition within the memory for storing secret data associated with and accessible by the end user application, the secure partition being inaccessible to the operating system image and other tasks operating on top of the secure platform.

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.